1. 上课约定须知

课程主题：HBase分布式NoSQL数据库 -- 第二次课 -- 源码解析
上课时间：20:00 - 23:00
课件休息：21:30 左右 休息10分钟
课前签到：如果能听见音乐，能看到画面，请在直播间扣 666 签到

2. 上次作业复盘

上次课程未布置作业!

3. 上次内容总结

```
1、如何设计一个分布式数据库头脑风暴
    如何提高查询效率？
        内存 + 磁盘
        内存数据良好的数据结构
        磁盘数据 + 布隆索引
        排序 ==> 二分查询 ==> 范围分区 + 索引 ===> 跳表
        读缓存 + 写缓存
    如何提高写入效率？
        WAL机制 + LSM-Tree存储引擎
2、HBase的架构设计
3、HBase的核心概念
    架构概念：HMaster + HRegionServer + Client + Zookeeper + HDFS
    表物理概念：rowkey + column family + qualifier + timestamp + value
    表逻辑概念：table + region + store + memstore + storeFile + blockcache
    核心用户动作概念：put + get/scan
```

```
    核心内部工作概念：flush + split + compact
4、HBase表模型
    四维表 value = get(rokwey, column family, qualifier, timestamp)
    多层map嵌套 table = map(rowkey, map(cf, map(qualifier, map(timestamp,
value))))
5、HBase的JavaAPI设计哲学
6、HBase的核心工作机制
    HBase的region定位/HBase的寻址机制
    HBase的写数据流程
    HBase的读数据流程
7、内部机制
    flush：hbase客户端写入数据到hbase服务端的时候，先写入memstore，然后flush到磁盘形成
storefile
    compact：当一个region的内部的storefile越来越多，就会执行compact动作把多个storefile合
并成一个
    split：当一个region的大小增长到10G的时候，这个region就会一分为二
```

额外的辅助知识：

```
1、怎么大批量插入数据：bulkload
2、怎么提高查询效率： filter coprocessor 各种缓存（读缓存） 各种索引（布隆索引）
3、调优： 建表调优：预分区
4、rowkey： 第一越短越好，第二防止数据热点（大量请求针对少量数据的查询：rowkey集中， 大量请求
针对大量数据来进行查询：rowkey分散）
```

4. 本次内容大纲

4.1. 本次内容概述

```
1、HBase集群启动脚本start-hbase.sh分析
2、HBase的HMaster启动流程解析
3、HBase的HRegionServer启动流程解析
```

源码的储备知识：

```
1、这个技术相关的核心流程的详细步骤， 大概的原理
2、根据你企业的实际情况和个人实际情况来决定技术的版本
3、搭建环境阅读环境
4、阅读的是分布式技术组件，序列化 + 网络通信
```

HBase 的 RPC

```
1、HBase-2.x 版本以前的 RPC 实现： nio + Prorobuf
2、HBase-2.x 版本以后的 RPC 实现： Netty
```

HBase的RPC相关的实现类：RpcServer（NettyRpcServer） + RpcClient（NettyRpcClient）

如果现在启动服务端（HMaster & HRegionServer）：最终肯定会有一个步骤，要启动 RPC 服务端
如果现在启动客户端（Connection ==> Admin | HTable）：要启动 RPC 客户端

继续发散：

```
1、HMaster 启动过程中，除了要启动 RPC 服务端以外，启动一些列服务，包括webui，参与选举 active master
2、HRegionServer 启动过程中，除了要启动 RPC 服务端以外，注册+心跳
```

Spark Flink 等技术的 Standalone 集群的启动，都有这些事儿！

ZK 的企业案例：使用分布式锁来进行主节点 active 角色选举！

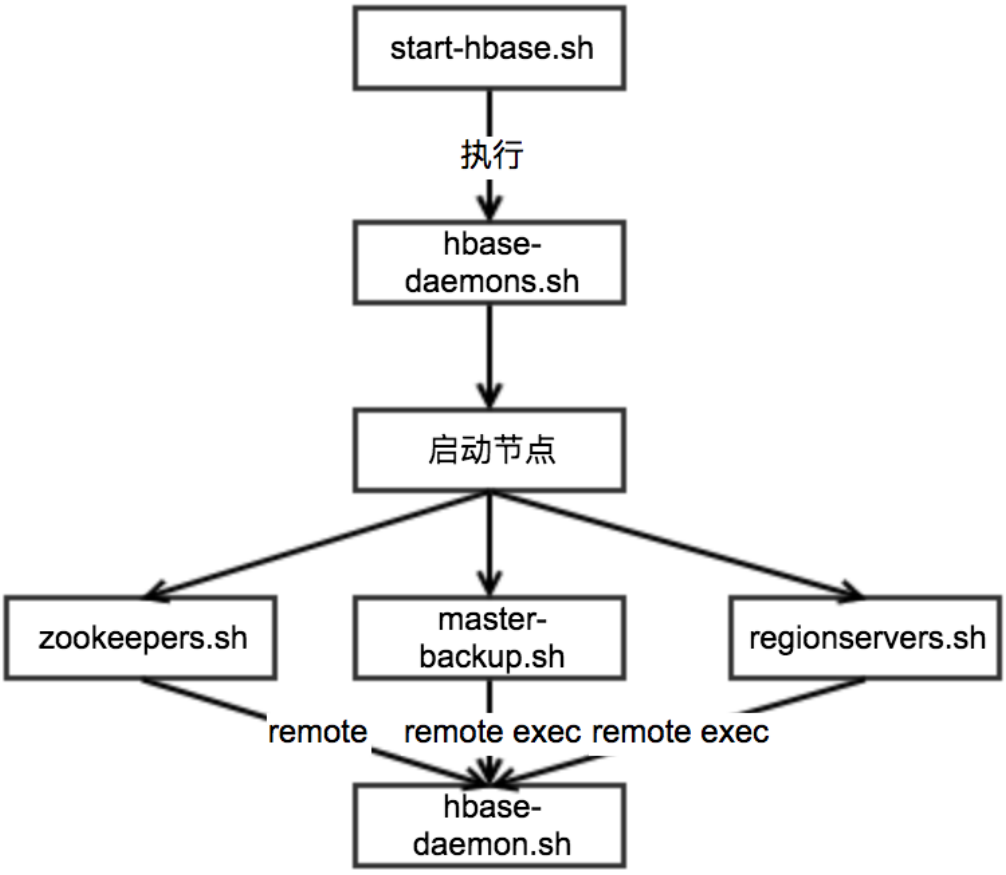
5. HBase源码分析

主体上大致分为三个大部分：

```
1、集群启动 Master RegionServer
2、put get 流程分析  createTable
3、flush  compact split 三大核心工作机制
```

5.1. HBase集群启动脚本start-hbase.sh分析

启动集群的shell命令：start-hbase.sh



```
# 启动：HMaster
hbase-daemon.sh start master
    java org.apache.hadoop.hbase.master.HMaster

# 启动：HRegionServer
hbase-daemon.sh start regionserver
    java org.apache.hadoop.hbase.master.HRegionSever
```

直接跳转到这两个类的 main 即可！

```
hbase shell
hbsae version
hbase start master|regionserver
```

5.2. HBase的HMaster启动流程解析

通过脚本分析得知，HMaster的启动入口：HMaster.main()

```
HMaster.main()
HMasterCommandLine.doMain()
HMasterCommandLine.run()

    # 启动 HMaster
    startMaster()

        # 第一步：通过反射构造 HMaster 实例
        # masterClass = HMaster.class
        HMaster master = HMaster.constructMaster(masterClass, conf);   // 进入构
造方法

            # 第一步：先调用 HRegionServer 的构造方法
            super(conf);
                # 创建 RPC 服务端
                rpcServices = createRpcServices();
                    createRpcServer(...)
                # 初始化和文件系统处理相关
                # HMaster 如果是第一次启动，那就必然会在HDFS上创建一系列初始化工作目录
                initializeFileSystem();
                # 创建 ZK 实例
                # HMaster 如果第一次启动，那就必然会在 zk 上创建一系列 znode 节点
                zooKeeper = new ZKWatcher()
                    # 获取一个 zk链接
                    recoverableZooKeeper = ZKUtil.connect(...)
                    # 通过这个 zk 链接创建各种基础 znode 节点
                    createBaseZNodes();
                # RPC 服务端启动：启动了 Scheduler , 启动了一堆 Handler
                this.rpcServices.start(zooKeeper);
                # 启动一个 杂务服务
                this.choreService = new ChoreService(getName(), true);
                # 启动一个 任务服务
                this.executorService = new ExecutorService(getName());
                # 启动 webui
                putUpWebUI();
```

```
            # 第二步：然后启动：初始化 ActiveMasterManager
            new ActiveMasterManager(zooKeeper, this.serverName, this);

        # 第二步：调用 HMaster 的 start() 启动
        master.start();                                                    // 进入
run()方法

            # 第一步：启动 webui 服务
            int infoPort = putUpJettyServer();

            # 第二步：启动 ActiveMasterManager
            startActiveMasterManager(infoPort);

                第一步：先注册 backup master 的信息到 zk 中
                MasterAddressTracker.setMasterAddress(...)

                第二步：去争抢分布式锁，成为 active master
                activeMasterManager.blockUntilBecomingActiveMaster(timeout,
status)
                    # 创建 master 节点下的 active master 节点
                    MasterAddressTracker.setMasterAddress(...)
                    # 再删除自己的 backup-master 信息
                    ZKUtil.deleteNodeFailSilent(this.watcher, backupZNode);

                第三步：完成成为 active master 之后的一些工作
                finishActiveMasterInitialization(status);
                    initializeMemStoreChunkCreator();   # 原来类似于 G1 垃圾回收期
                    new MasterFileSystem(conf);
                    new MasterWalManager(this);
                    ZKClusterId.setClusterId(hbase.clusterID);
                    createServerManager(this);  # 管理 RS
                    new SplitWALManager(this);   # 负责进行 Split 动作的一个组件
                    createProcedureExecutor();
                    createAssignmentManager(this);  # 负责管理 region 的分派工作
                    new TableStateManager(this);  # 管理hbase的表的状态 disable
offline
                    InitMetaProcedure temp = new InitMetaProcedure();
                                procedureExecutor.submitProcedure(temp);
                    new FavoredNodesManager(this);
                    ....
                    startServiceThreads();
                    waitForRegionServers(status);   # 等待 RS 注册
                    waitForMetaOnline()             # 等待meta表上线
```

HMaster 的启动到此为止!

5.3. HBase的HRegionServer启动流程解析

通过脚本分析得知，HRegionServer 的启动入口：HRegionServer.main()

```
HRegionServer.main()
HRegionServerCommandLine.run()
    hrs = HRegionServer.constructRegionServer(...);   // HRegionServer 的构造方法
        rpcServices = createRpcServices();
```

```
        initializeFileSystem();
        zooKeeper = new ZKWatcher()
        this.rpcServices.start(zooKeeper);
        this.executorService = new ExecutorService(getName());
        this.choreService = new ChoreService(getName(), true);
        putUpWebUI();
    hrs.start();                                        // HRegionServer 的run()
方法
        # 初始化
        preRegistrationInitialization();
        # 注册
        reportForDuty();
        # 注册响应处理
        handleReportForDutyResponse(w);
        # 心跳汇报 regioin的状态给master
        tryRegionServerReport(lastMsg, now);
```

6. 本次课程总结

本次课程的主要内容，就是给大家讲解 HBase 的集群启动源码剖析，从中，HBase 集群在启动的时候，会做哪些事情。对于一个数据库系统来说，启动的时候，会启动一些必要的服务，会恢复数据的状态等等。

7. 本次课程作业

当时学完 Zookeeper 的时候，给大家布置了作业，让各位实现一个类似于 ZooKeeper 的数据模型系统，具备一些基本的功能，比如：

```
1、通过树形方式来组织数据，具备基本的节点增删改查的功能
2、该系统具备冷启动恢复数据状态的功能。
```

现在来实现一个 HBase 存储系统！两点要求：

```
1、设计一个系统存储key_value类型数据，该系统为四维表模型，让系统具备根据一个条件，两个条件，三个条件，四个条件查找value的能力，如果不是根据四个条件来查找，返回单个value，如果不是根据四个条件来查询，则返回value的集合，最好不要只是value的集合，最好是一个map。
2、该系统可以冷启动恢复数据状态
```